10

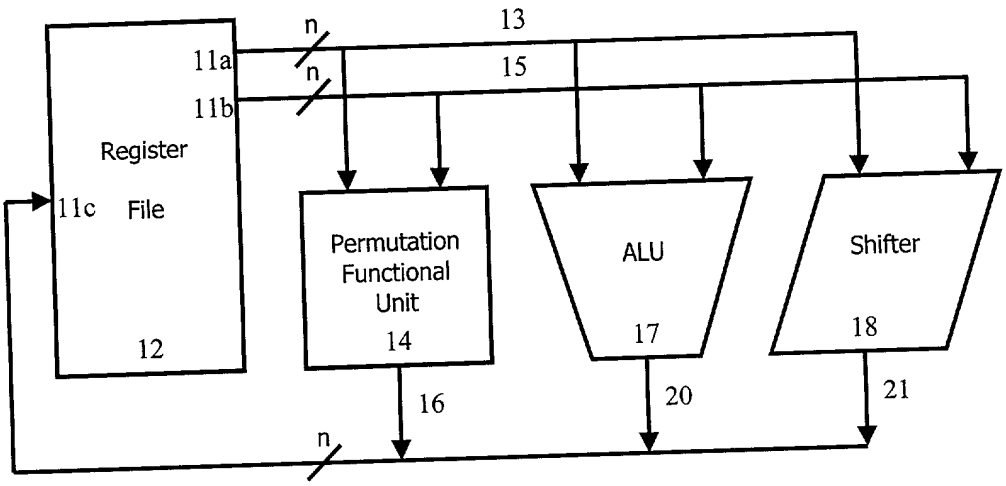
n
13
11a
n
15
11b
Register
File
11c
12
Permutation
Functional
Unit
14
ALU
17
Shifter
18
16
20
21
n

Fig. 1

22

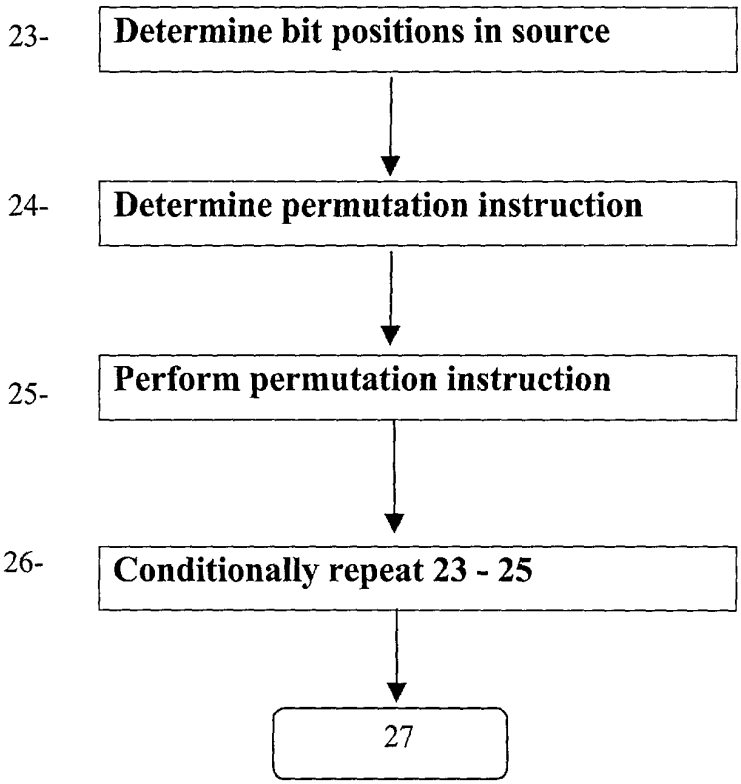

Fig. 2

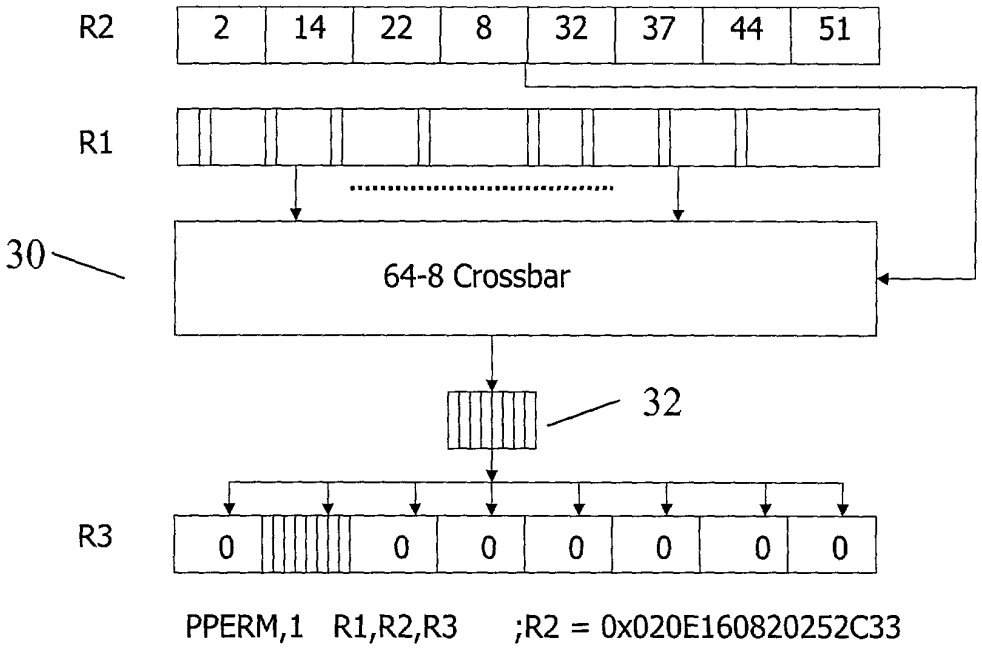
R2
2
14
22
8
32
37
44
51
R1
30
64-8 Crossbar
32
R3
0
0
0
0
0
0
0
PPERM,1 R1,R2,R3 ;R2 = 0x020E160820252C33

Fig. 3A

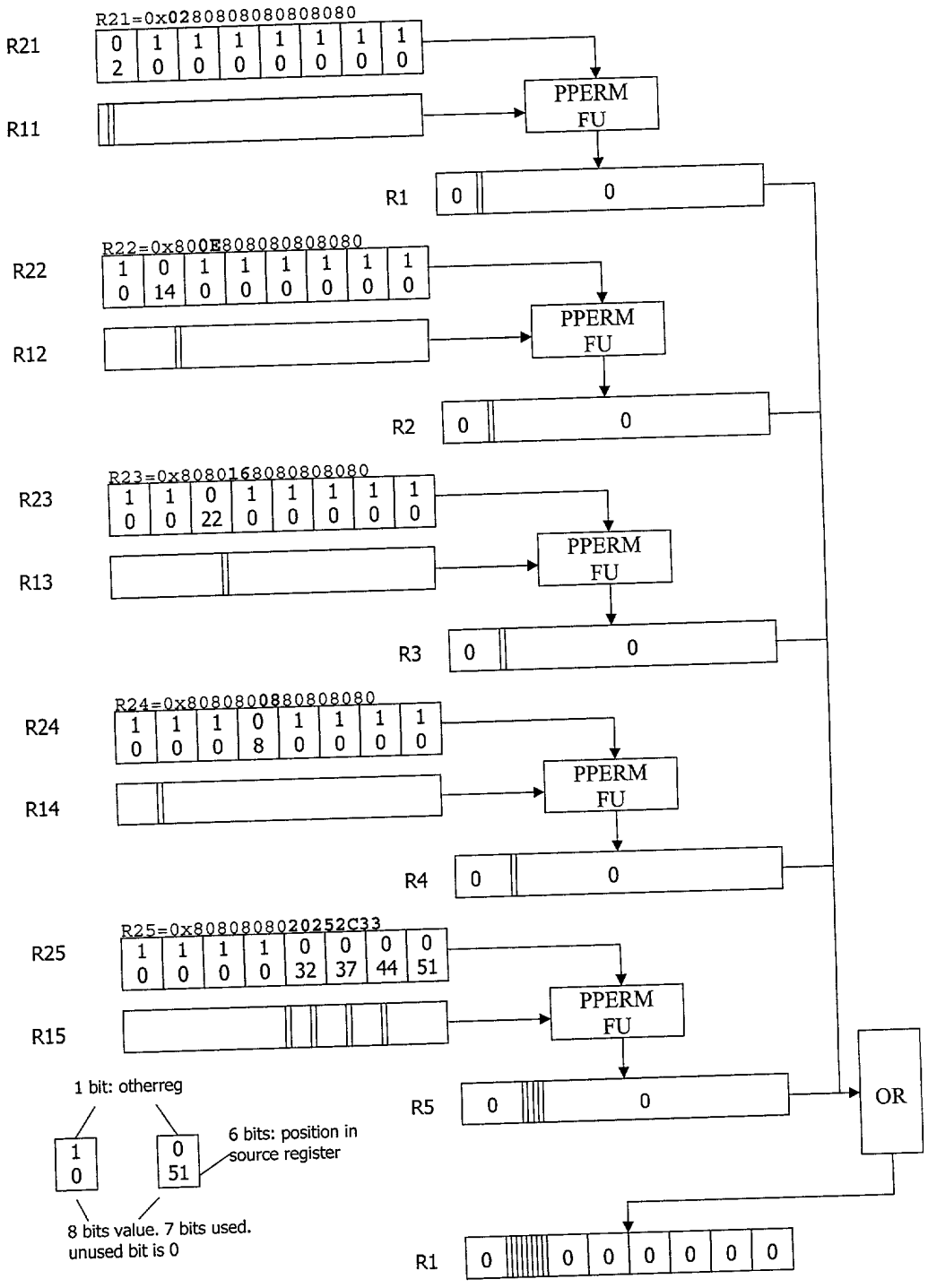
R21=0x0280808080808080
R21
R11
PPERM FU
R1
R22=0x800E808080808080
R22
R12
PPERM FU
R2
R23=0x8080168080808080
R23
R13
PPERM FU
R3
R24=0x8080800880808080
R24
R14
PPERM FU
R4
R25=0x8080808020252C33
R25
R15
PPERM FU
R5
OR
R1
1 bit: otherreg
6 bits: position in source register
8 bits value. 7 bits used. unused bit is 0

Fig. 3B

100

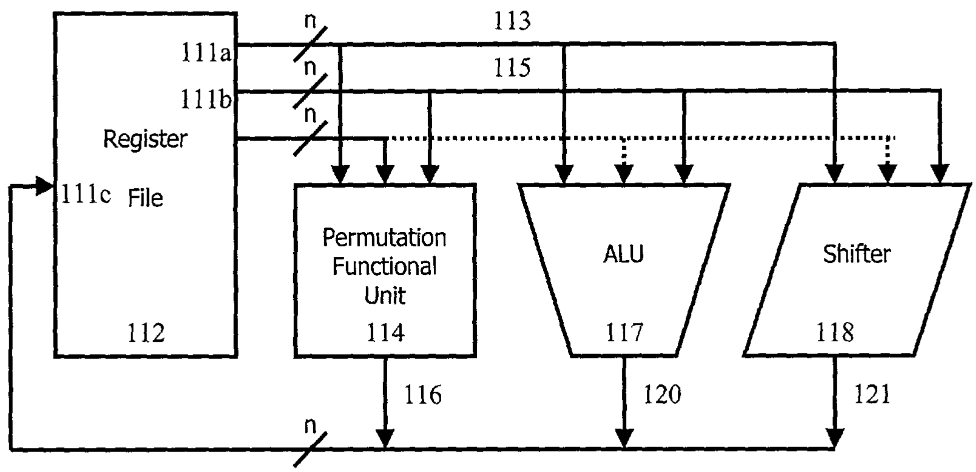

Fig. 4A

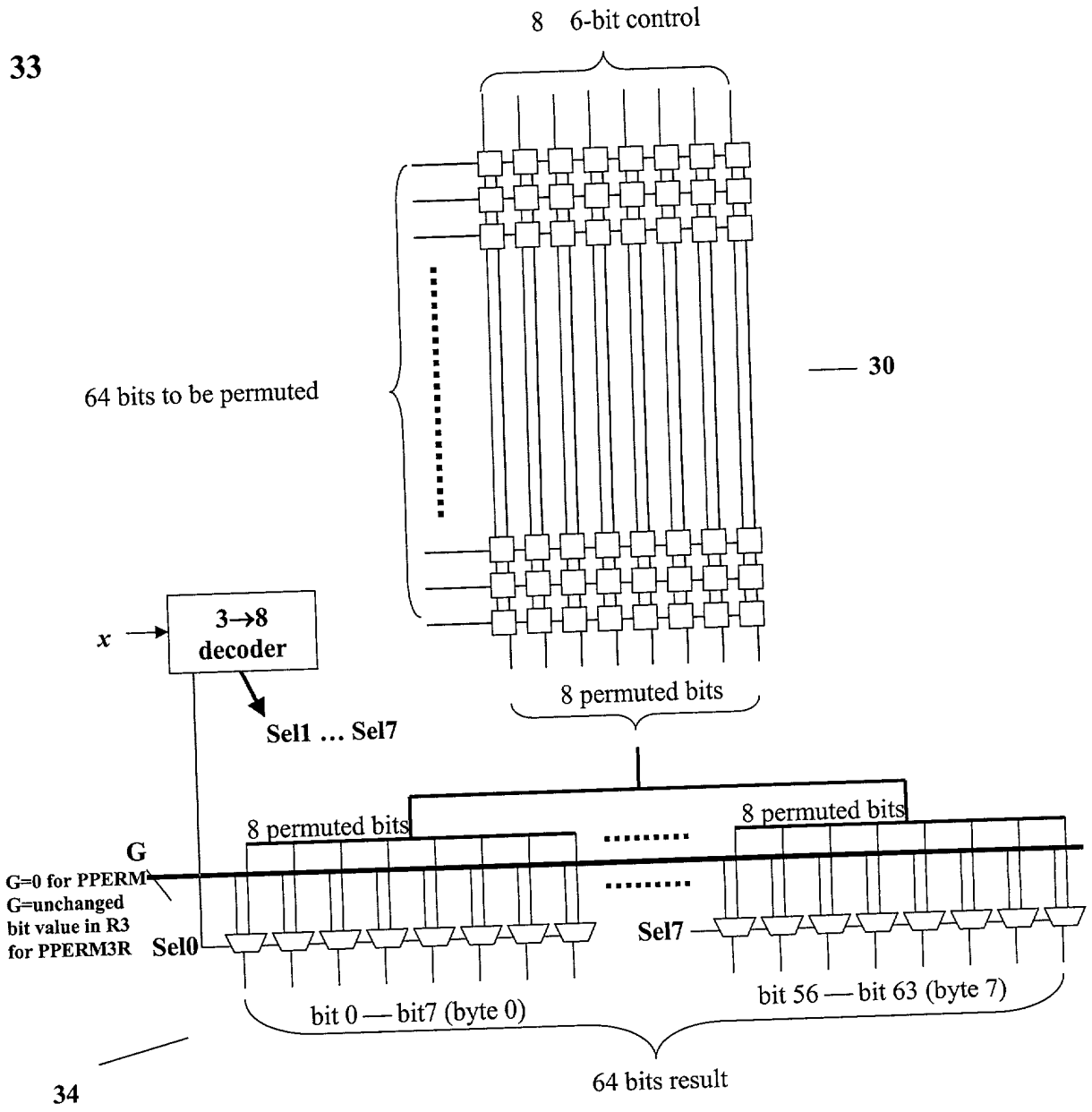
33
8 6-bit control
64 bits to be permuted
30
3→8 decoder
x
Sel1 … Sel7
8 permuted bits
8 permuted bits
8 permuted bits
G
G=0 for PPERM
G=unchanged bit value in R3 for PPERM3R
Sel0
Sel7
bit 0 — bit7 (byte 0)
bit 56 — bit 63 (byte 7)
64 bits result
34

Fig. 4B

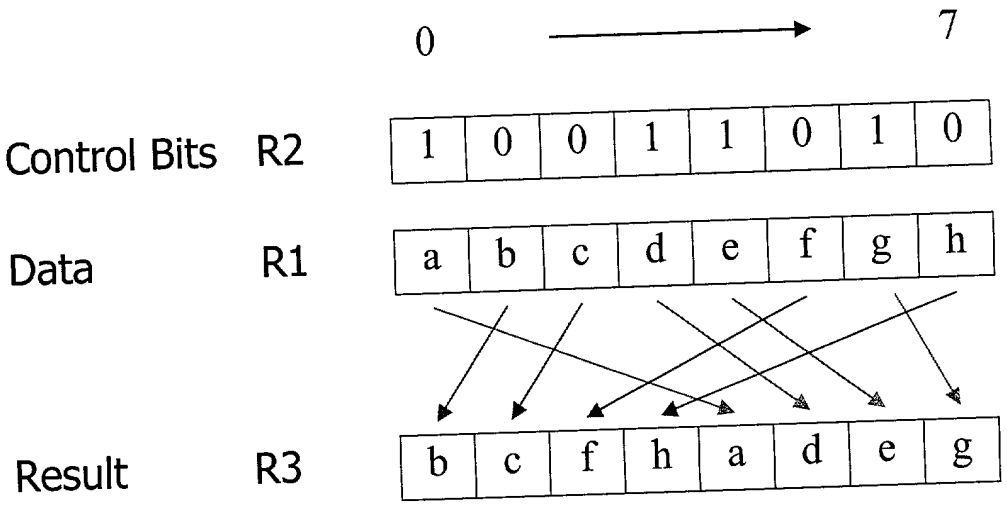
0
7
Control Bits R2
1 0 0 1 1 0 1 0
Data R1
a b c d e f g h
Result R3
b c f h a d e g

Fig. 5

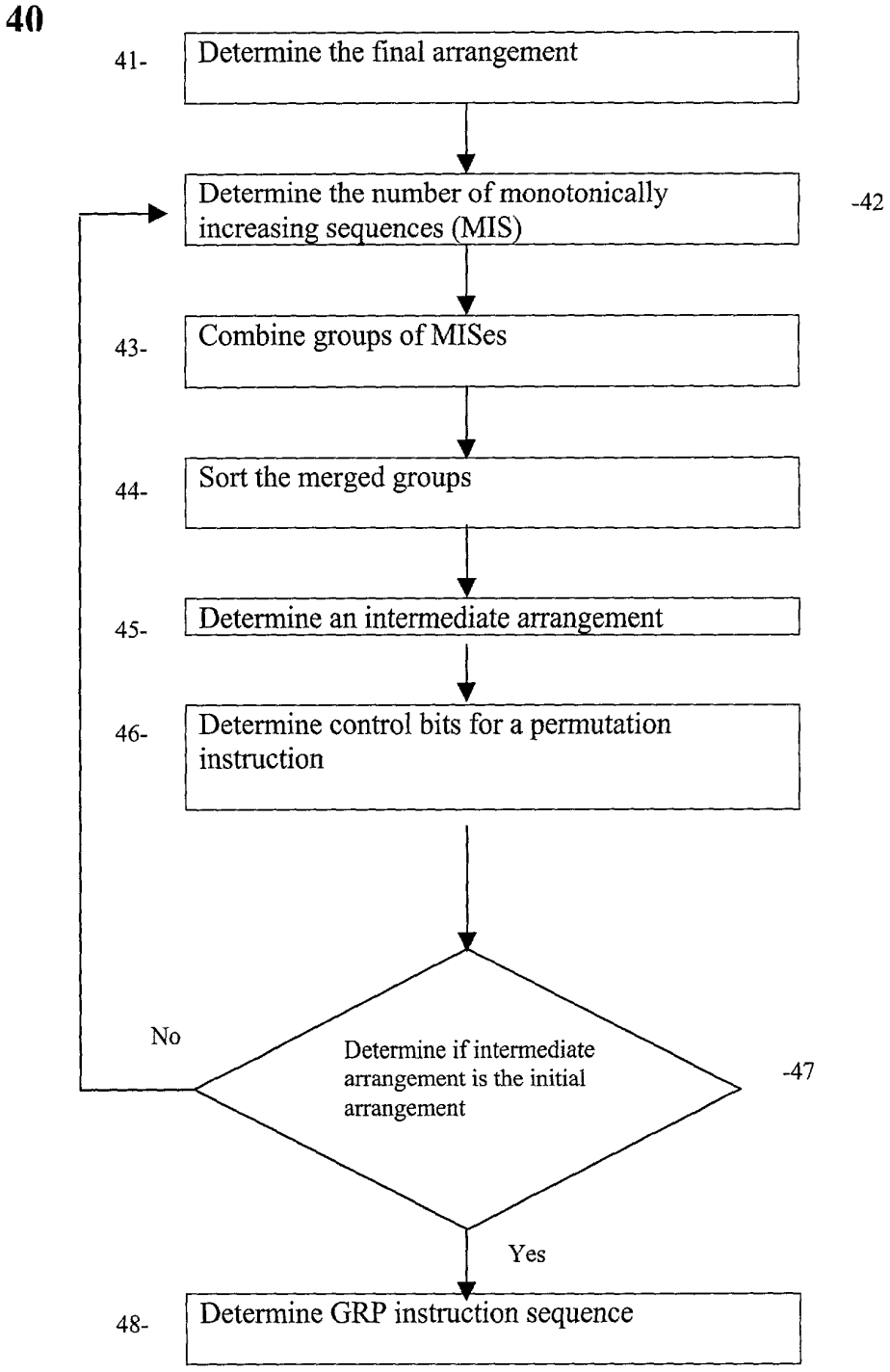
40
41-
Determine the final arrangement
Determine the number of monotonically increasing sequences (MIS)
-42
43-
Combine groups of MISes
44-
Sort the merged groups
45-
Determine an intermediate arrangement
46-
Determine control bits for a permutation instruction
No
Determine if intermediate arrangement is the initial arrangement
-47
Yes
48-
Determine GRP instruction sequence

Fig. 6

50

51

		Iteration 1	Iteration 2
52-	P	(5, 0, 1, 2, 4, 3, 7, 6)	(3, 5, 7, 0, 1, 2, 4, 6)
53-	MISes in P	(5)(0, 1, 2, 4)(3, 7)(6)	(3, 5, 7), (0, 1, 2, 4, 6)
54-	Combining MISes	(5, 3, 7),(0, 1, 2, 4, 6)	(3, 5, 7, 0, 1, 2, 4, 6)
55-	Sorting	(3, 5, 7), (0, 1, 2, 4, 6)	(0, 1, 2, 3, 4, 5, 6, 7)
56-	New Arrangement	(3, 5, 7, 0, 1, 2, 4, 6)	(0, 1, 2, 3, 4, 5, 6, 7)
57-	Control bits for GRP instruction	(1, 0, 1, 0, 0, 0, 0, 1)	(1, 1, 1, 0, 1, 0, 1, 0)

Fig. 7

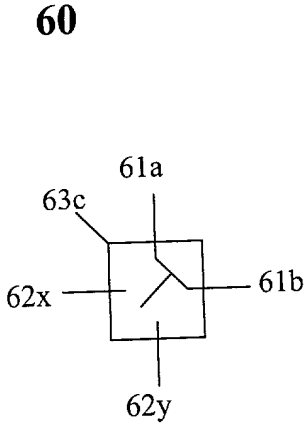

Fig. 8A

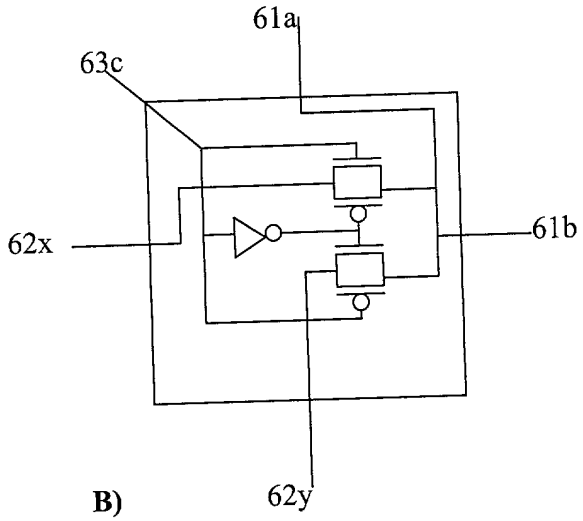

Fig. 8B

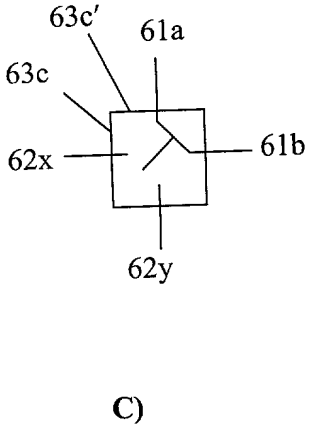

Fig. 8C

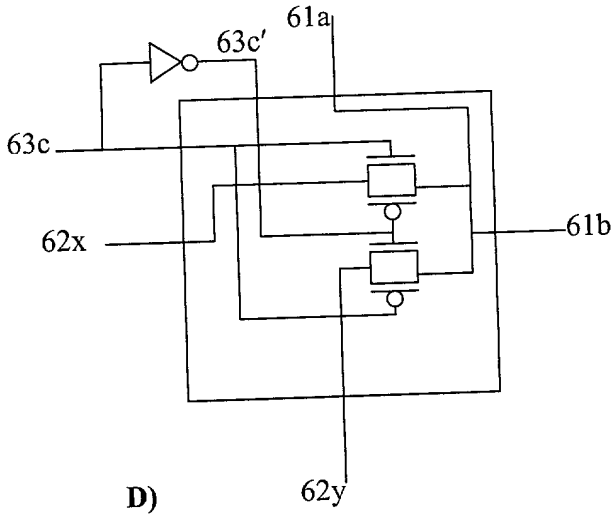

Fig. 8D

68

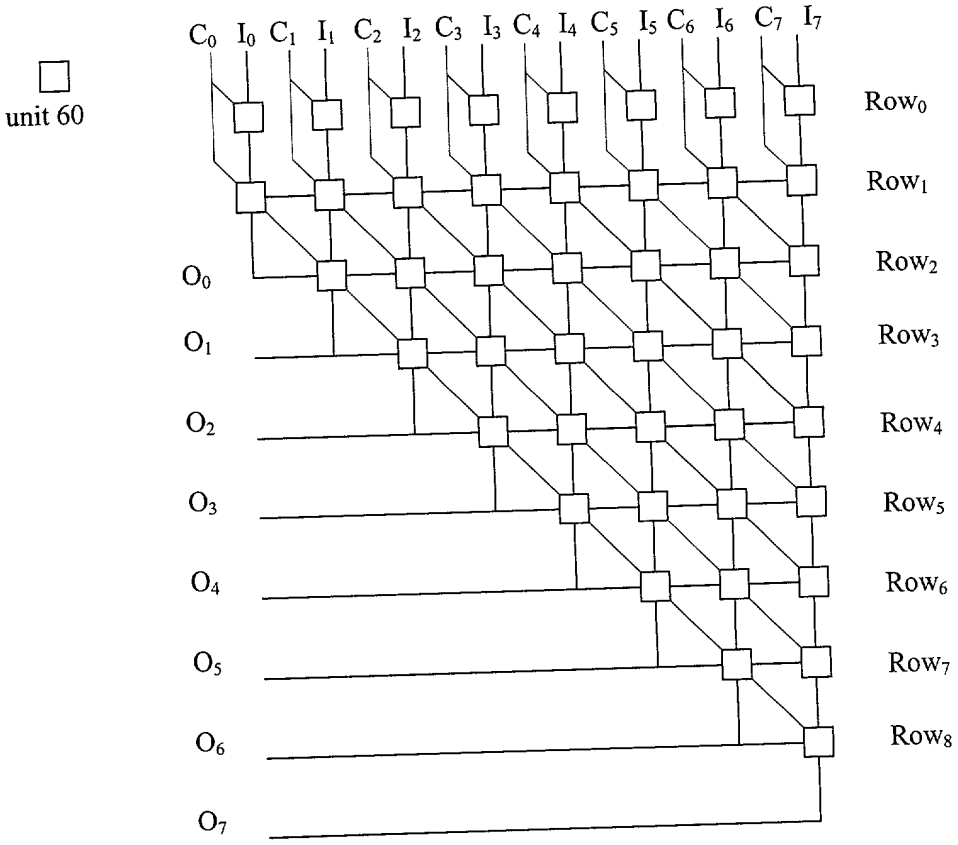

Fig. 9A

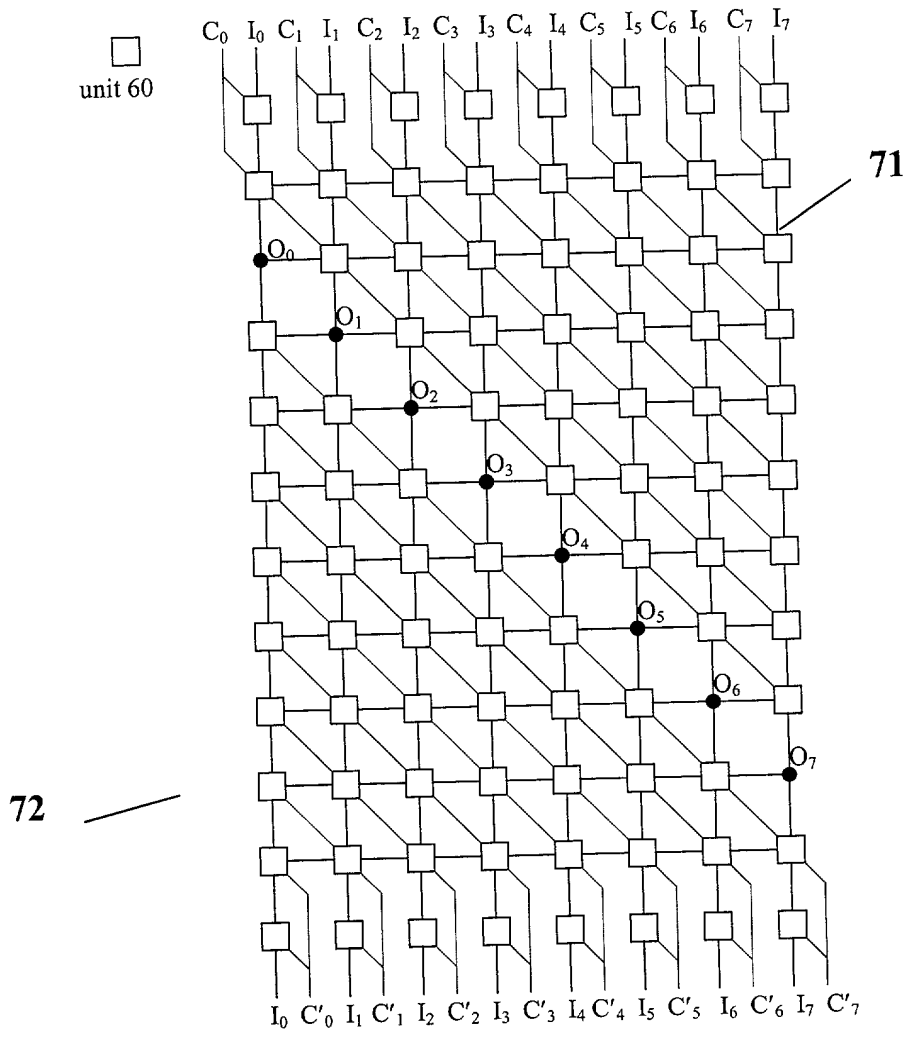
70
unit 60
71
72
C_0 I_0 C_1 I_1 C_2 I_2 C_3 I_3 C_4 I_4 C_5 I_5 C_6 I_6 C_7 I_7
O_0
O_1
O_2
O_3
O_4
O_5
O_6
O_7
I_0 C'_0 I_1 C'_1 I_2 C'_2 I_3 C'_3 I_4 C'_4 I_5 C'_5 I_6 C'_6 I_7 C'_7

Fig. 9B

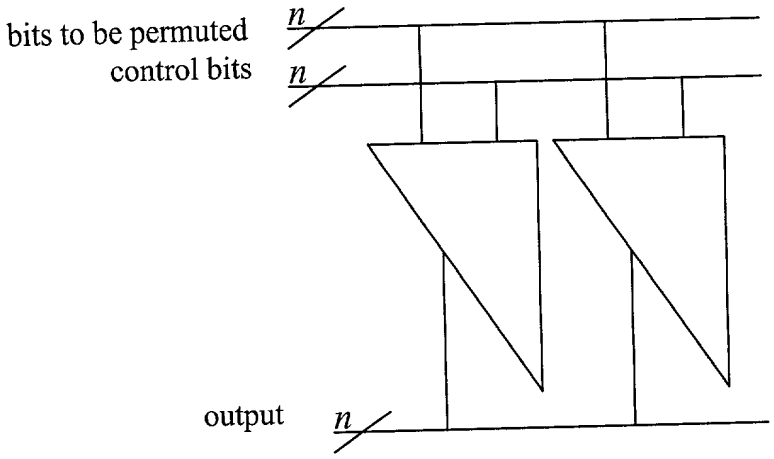

Fig. 10

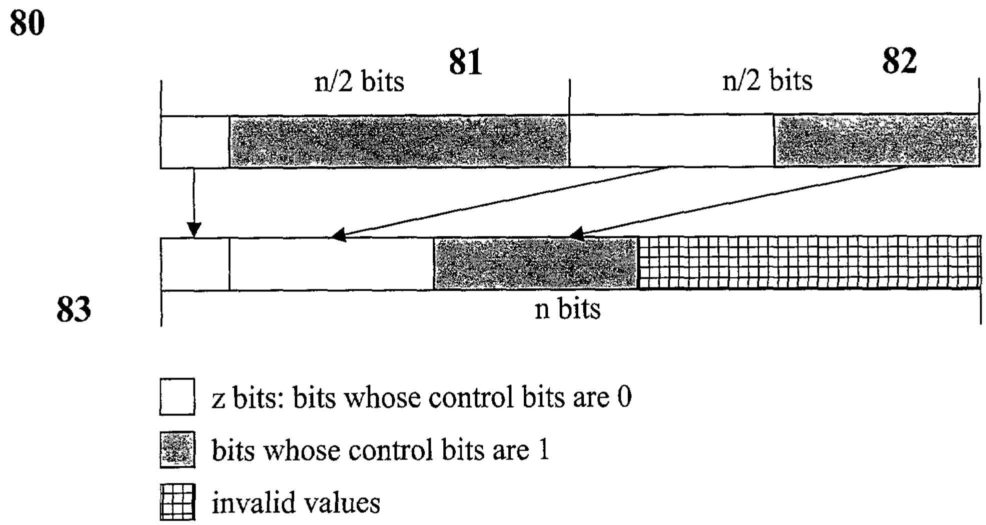
80
n/2 bits
81
n/2 bits
82
83
n bits
z bits: bits whose control bits are 0
bits whose control bits are 1
invalid values

Fig. 11

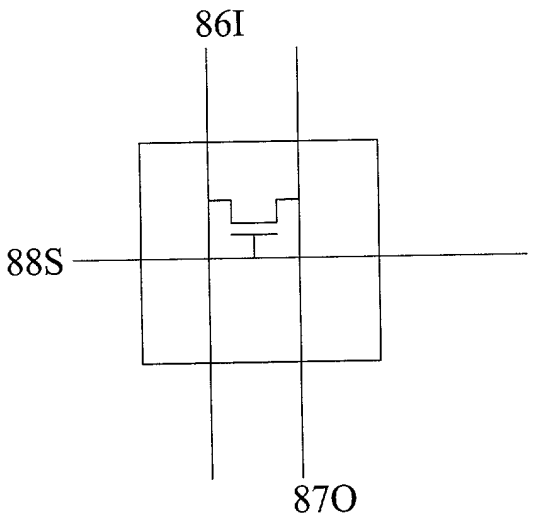

Fig. 12

90

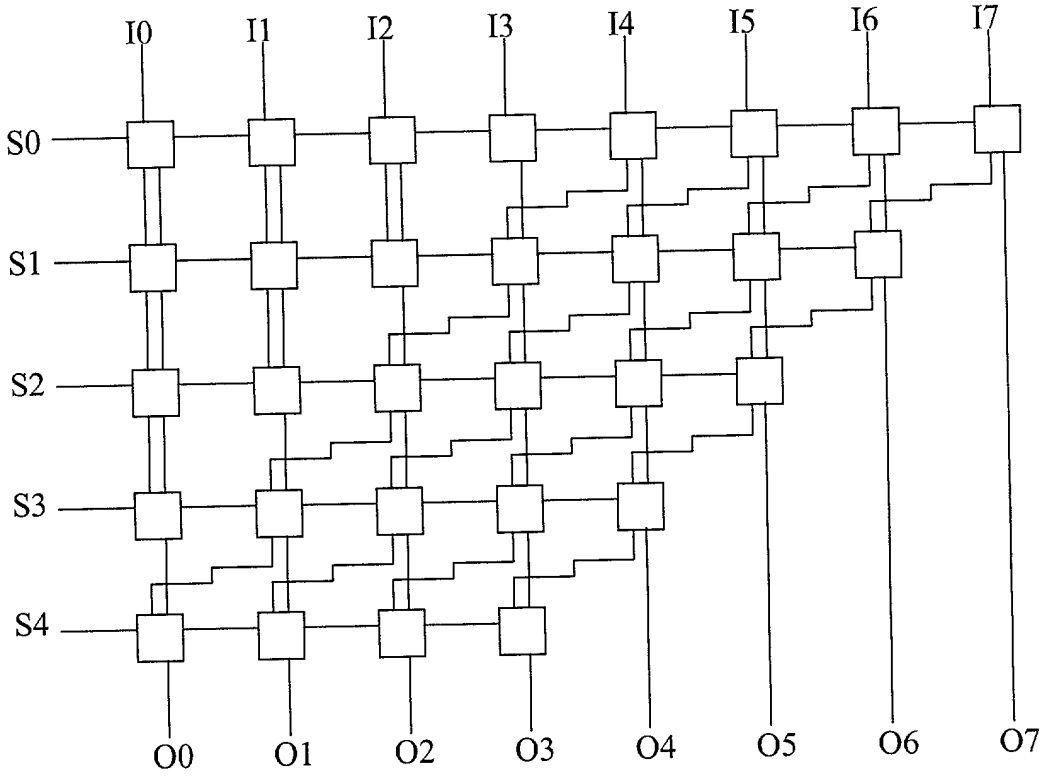

Fig. 13

92

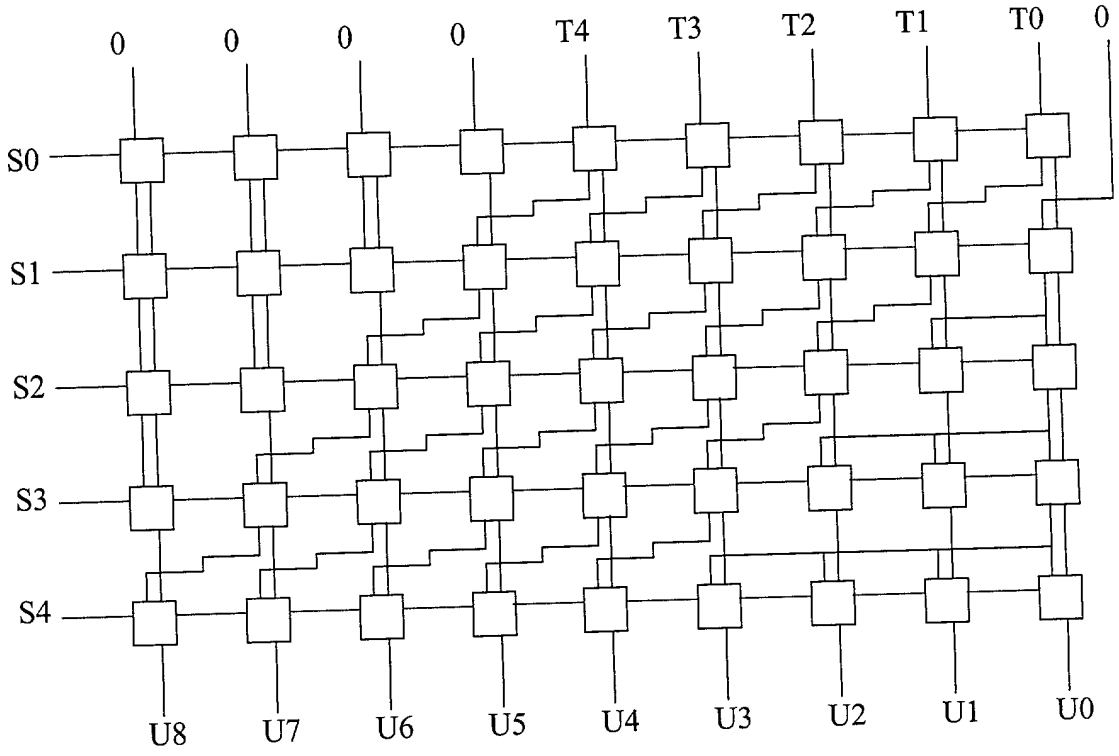

Fig. 14

94

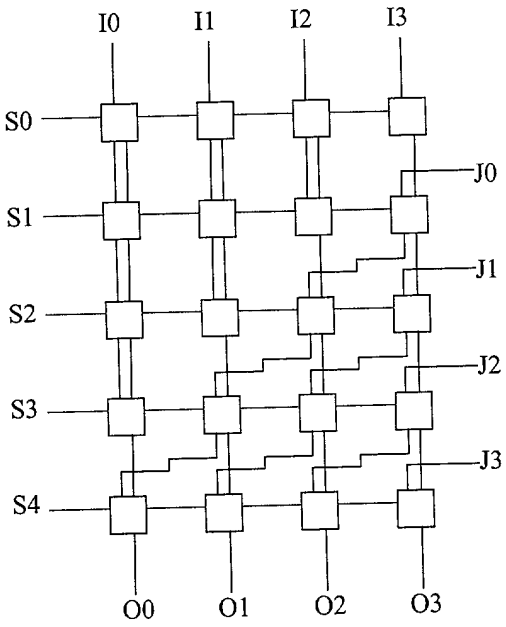

Fig. 15

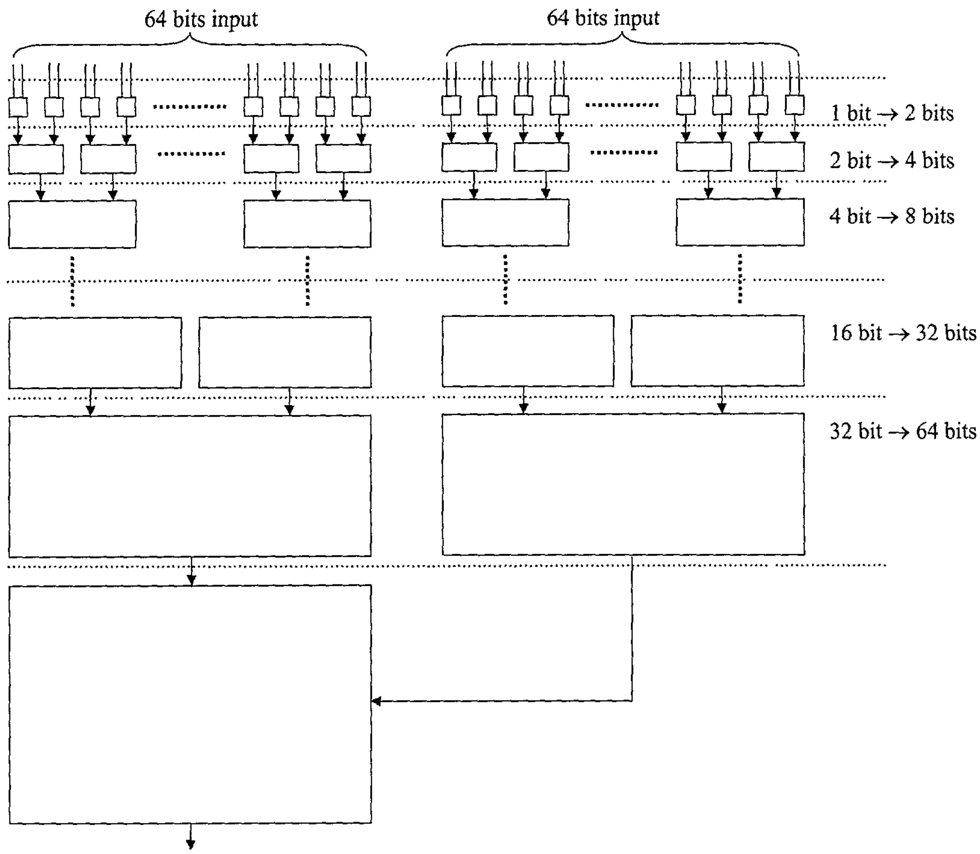

Fig. 16

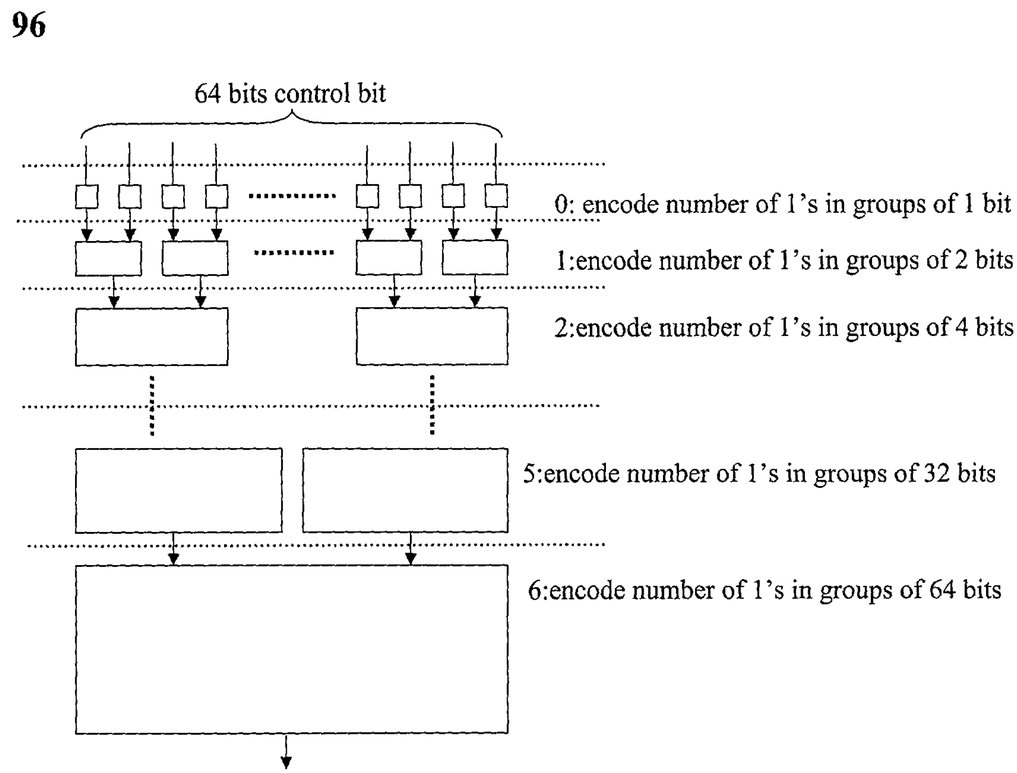
96
64 bits control bit
0: encode number of 1's in groups of 1 bit
1:encode number of 1's in groups of 2 bits
2:encode number of 1's in groups of 4 bits
5:encode number of 1's in groups of 32 bits
6:encode number of 1's in groups of 64 bits

Fig. 17

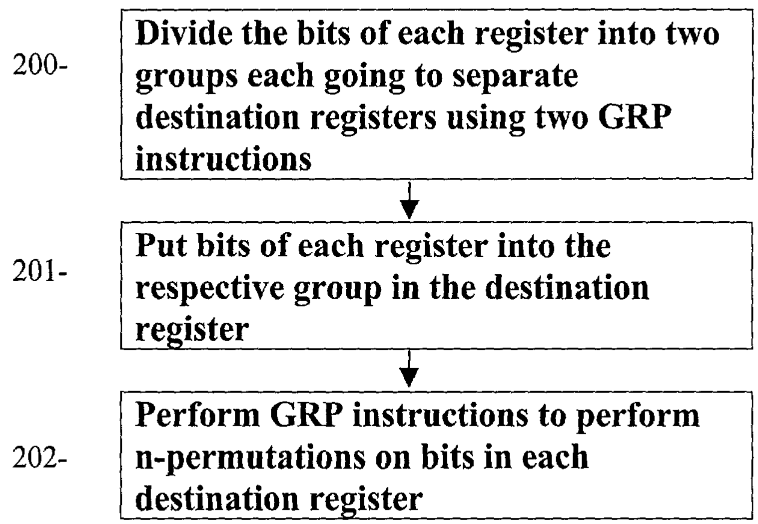
200-
Divide the bits of each register into two groups each going to separate destination registers using two GRP instructions
201-
Put bits of each register into the respective group in the destination register
202-
Perform GRP instructions to perform n-permutations on bits in each destination register

Fig. 18A

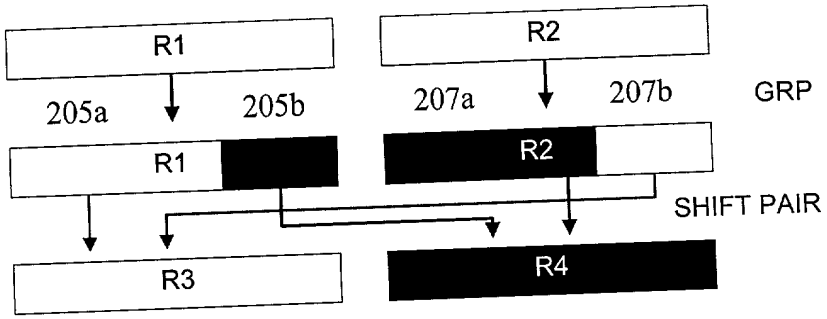

Fig. 18B

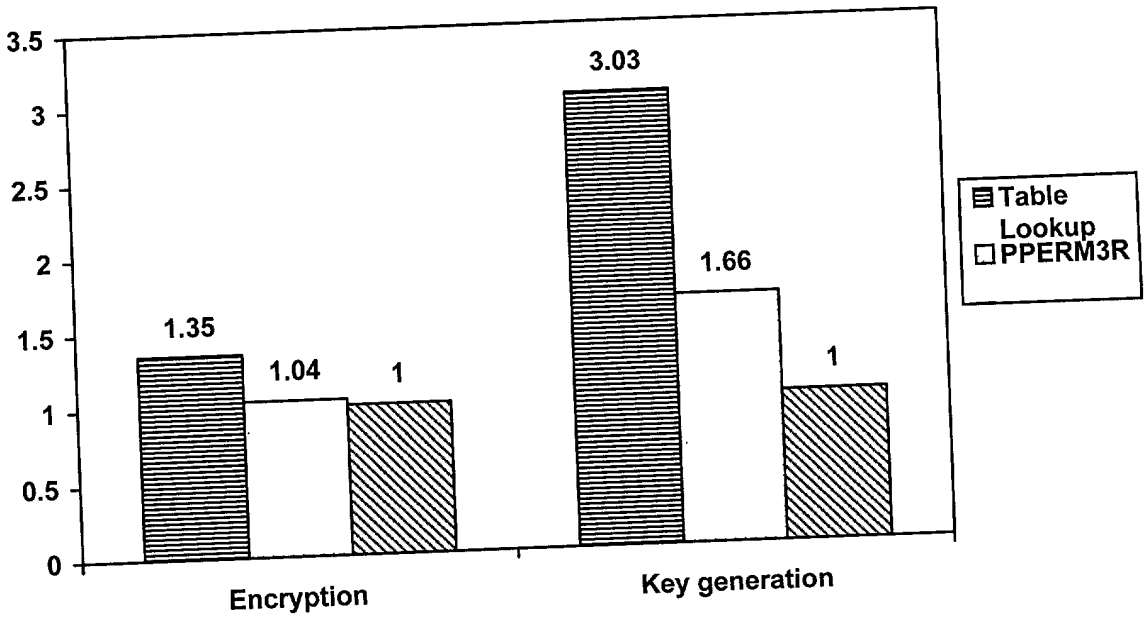
3.5
3
2.5
2
1.5
1
0.5
0
1.35
1.04
1
3.03
1.66
1
Encryption
Key generation
Table Lookup
PPERM3R

Fig. 19

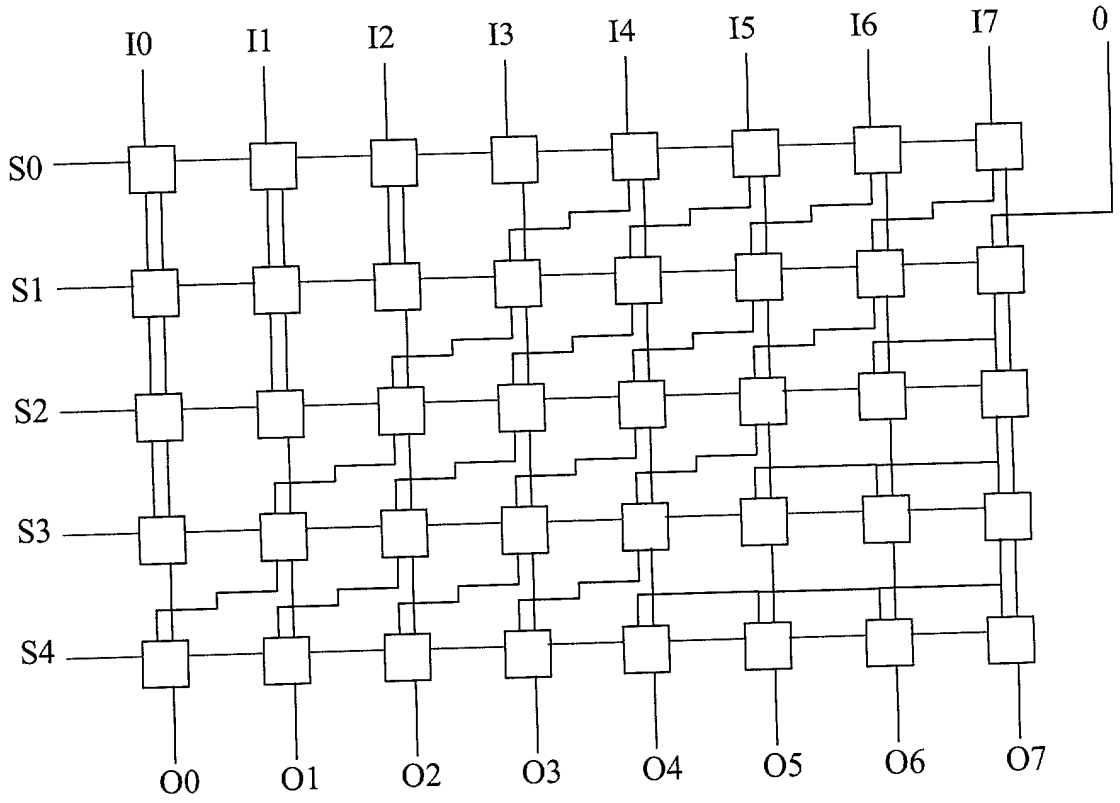

Fig. 20

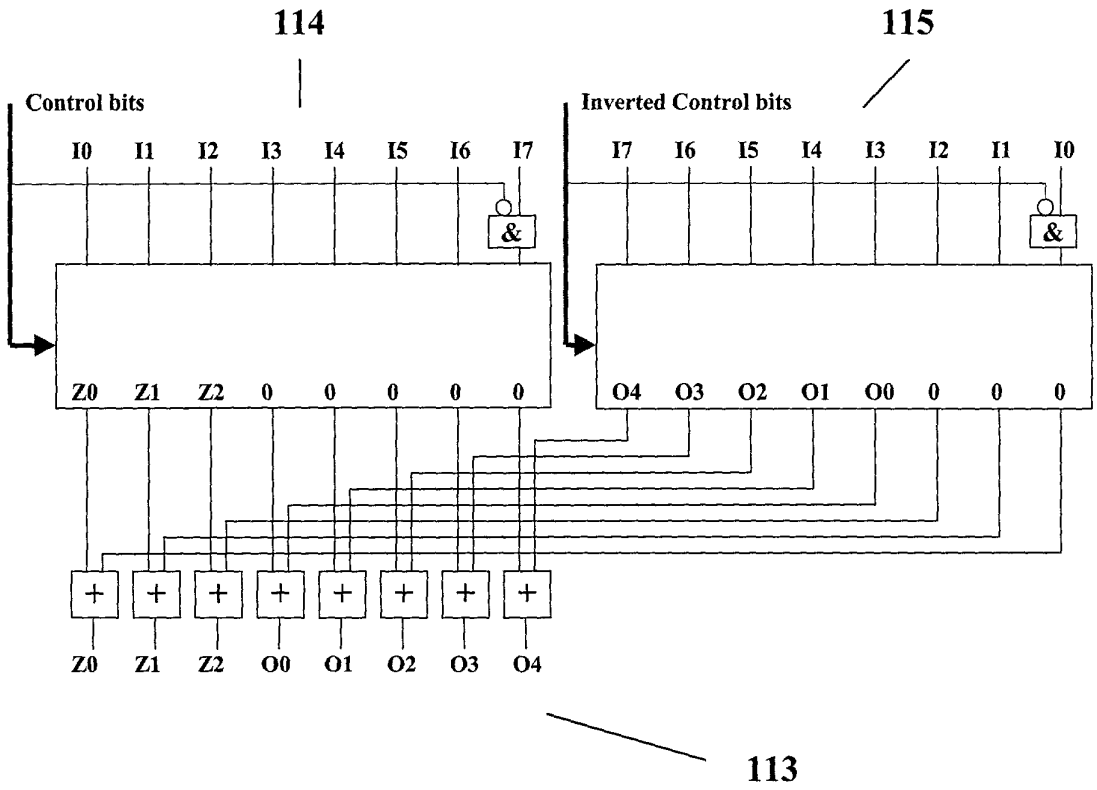
114
115
Control bits
Inverted Control bits
I0 I1 I2 I3 I4 I5 I6 I7
I7 I6 I5 I4 I3 I2 I1 I0
&
&
Z0 Z1 Z2 0 0 0 0 0
O4 O3 O2 O1 O0 0 0 0
+ + + + + + + +
Z0 Z1 Z2 O0 O1 O2 O3 O4
113

Fig. 21

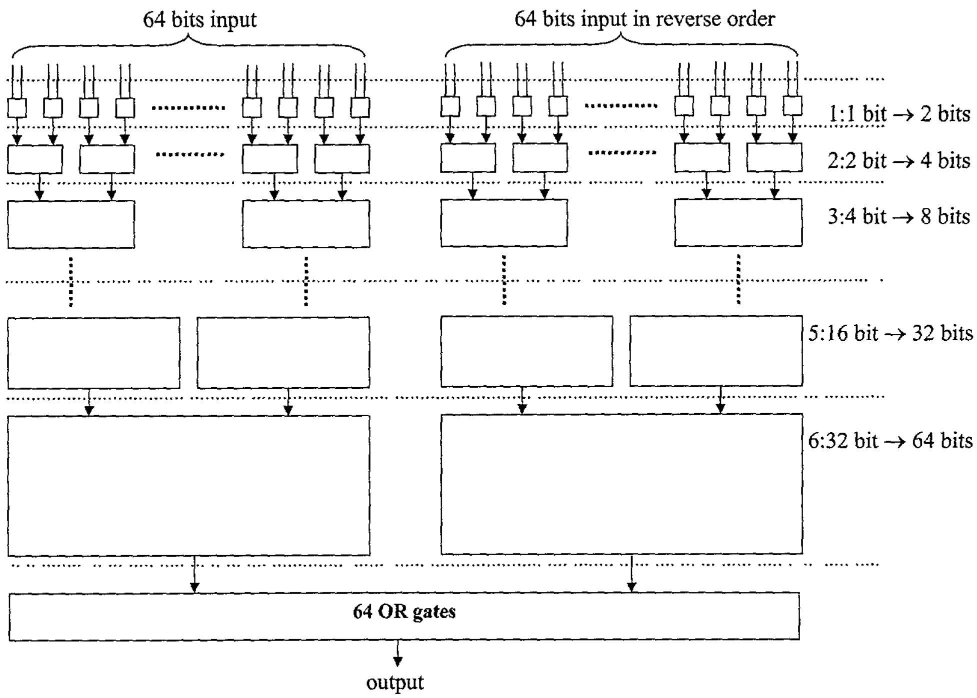

Fig. 22

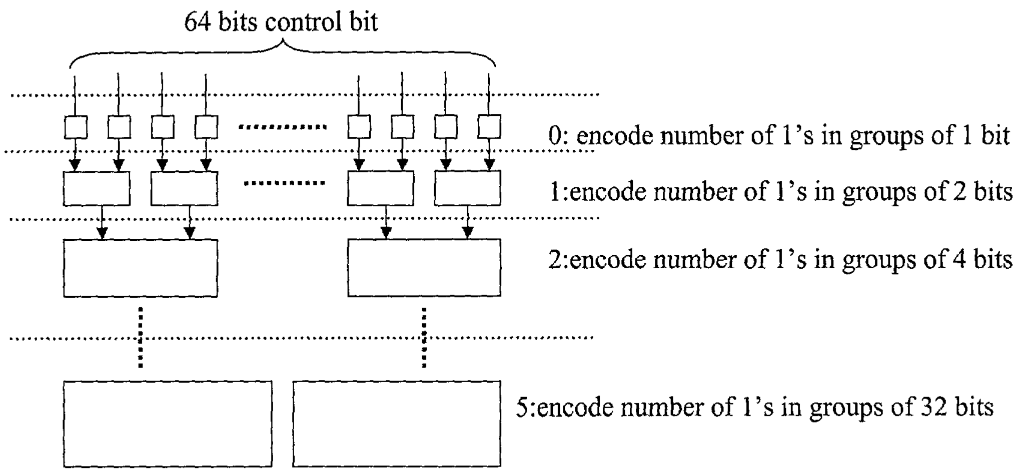
64 bits control bit
0: encode number of 1's in groups of 1 bit
1:encode number of 1's in groups of 2 bits
2:encode number of 1's in groups of 4 bits
5:encode number of 1's in groups of 32 bits

Fig. 23